LHC Data Analysis Using NFSv4.1 (pNFS):

A Detailed Evaluation

-Short introduction into dCache and NFSv4.1 (pNFS)

-First simple tests

-ATLAS HammerCloud

-CMS Analysis

-Reading ROOT files

Yves Kemp, Dmitry Ozerov, (DESY IT)
Tigran Mkrtchyan, Patrick Fuhrmann (dcache.org)
Johannes Elmsheuser (LMU Munich), Hartmut Stadie (Uni Hamburg)

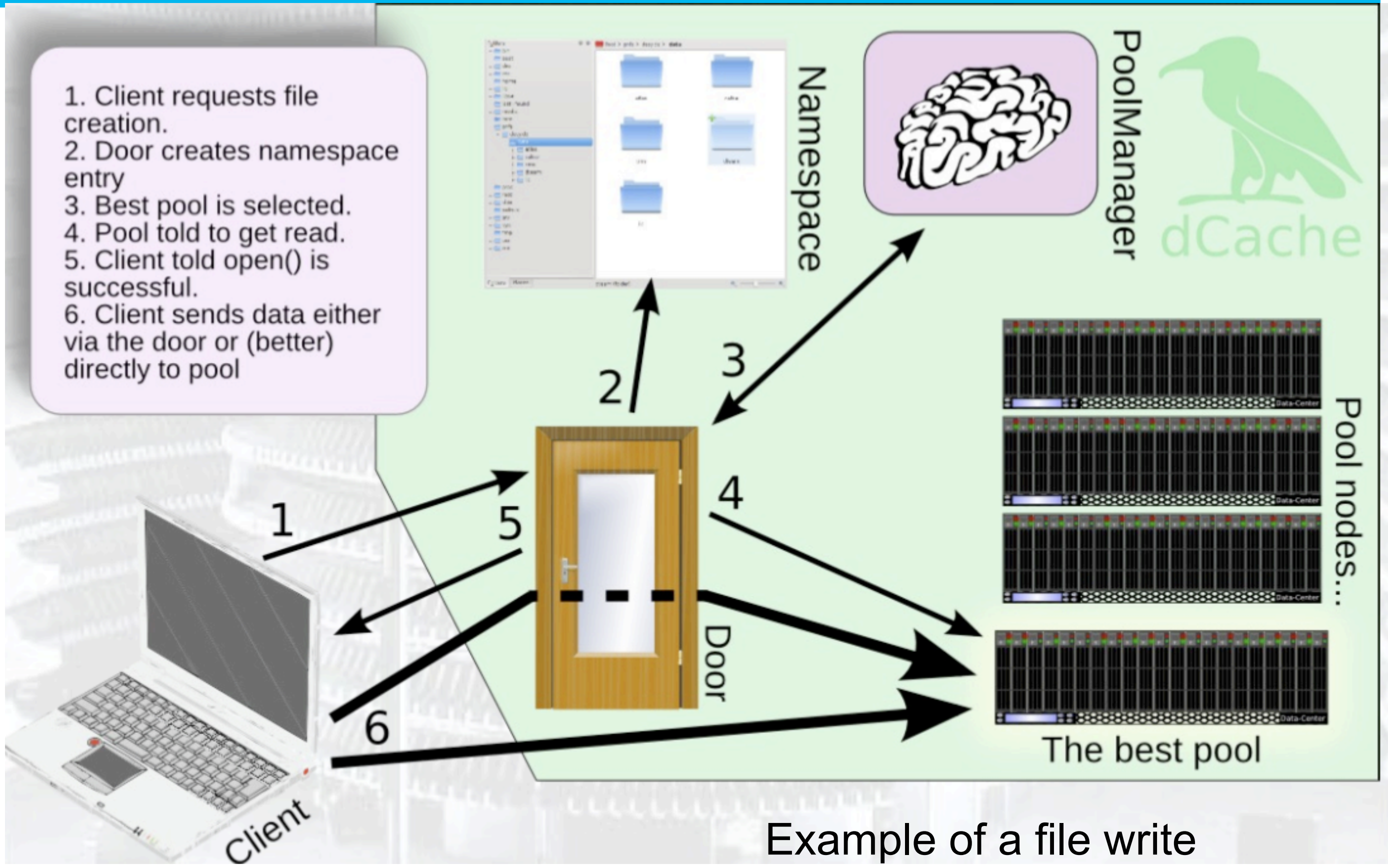
Taipei, 10/20/2010 CHEP 2010

HELMHOLTZ
ASSOCIATION

DESY

dCache in a nutshell

- Storage system, developed at DESY, FNAL and NDGF
- Objects stored: Files
- Files in pools, pools on poolnodes, many of them
- Client connects to a door, which speaks the desired protocol
- At the end the file is transferred directly between pool and client



1. Client requests file creation.
2. Door creates namespace entry
3. Best pool is selected.
4. Pool told to get read.
5. Client told open() is successful.
6. Client sends data either via the door or (better) directly to pool

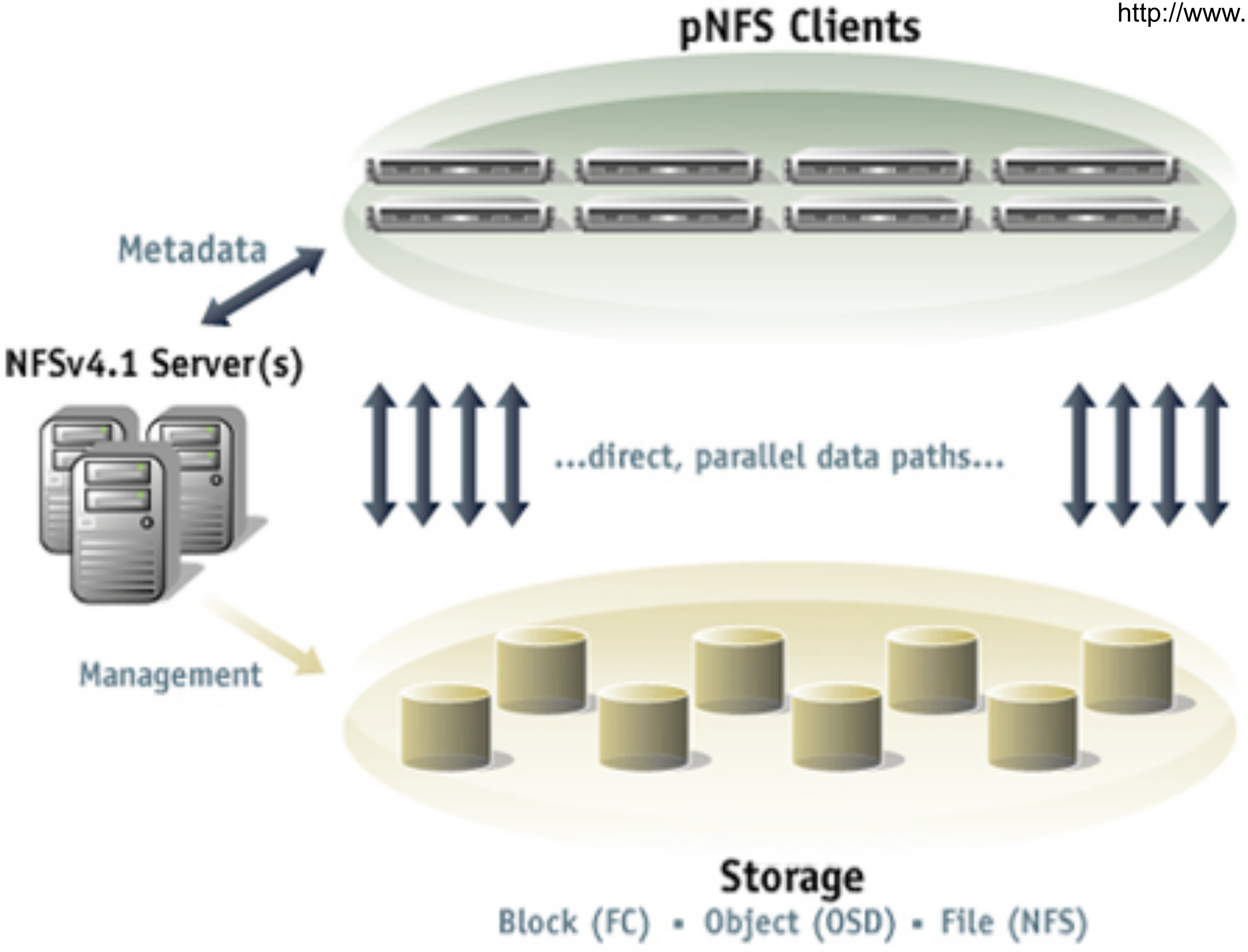
Example of a file write

In reality a little bit more complicated
Many talks and posters around dCache at CHEP

Check http://www.dcache.org/

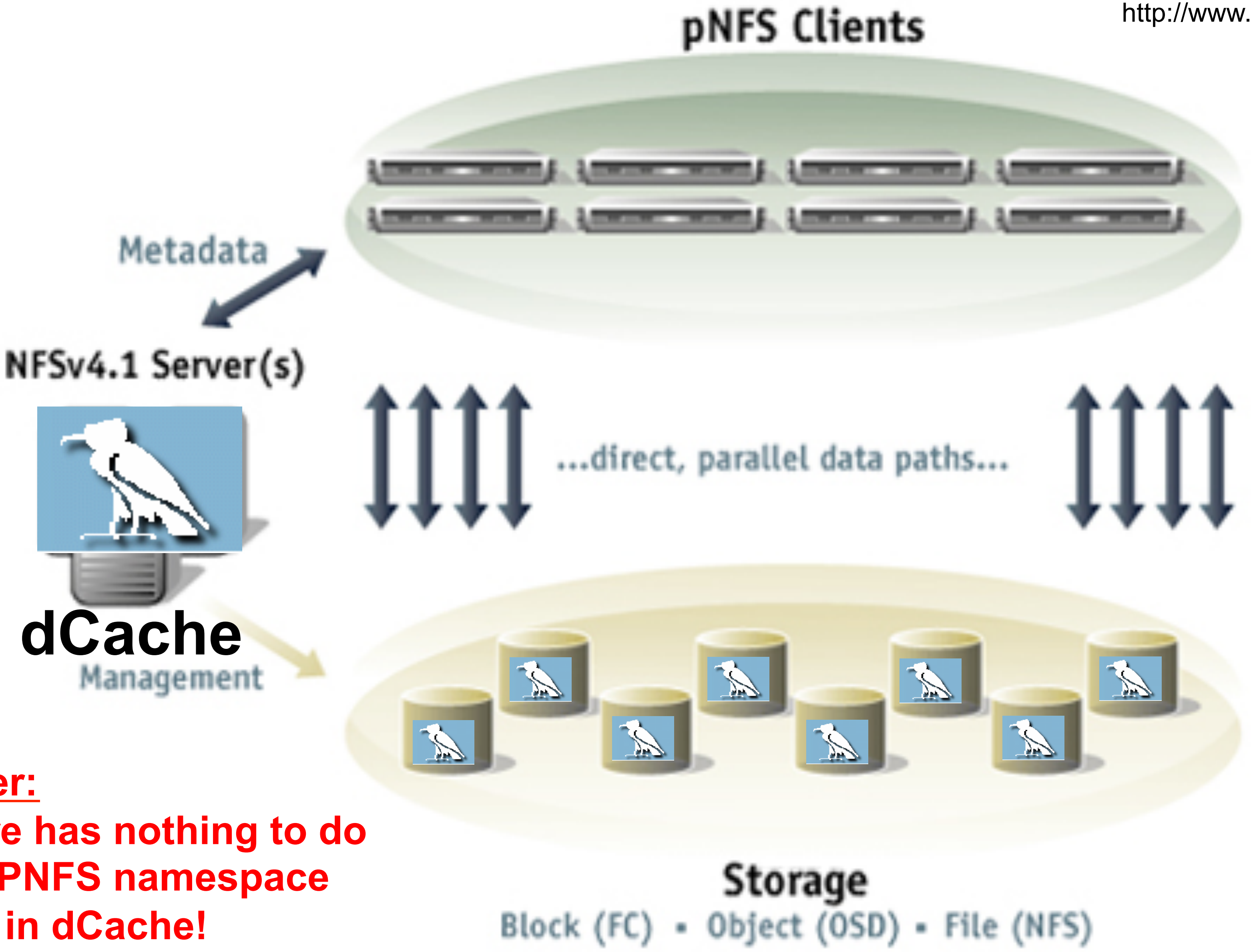
NFS v4.1 / pNFS from the infrastructure view

http://www.pnfs.com/


pNFS Clients
Metadata
NFSv4.1 Server(s)
...direct, parallel data paths...
Management
Storage
Block (FC) • Object (OSD) • File (NFS)

NFS v4.1 / pNFS from the infrastructure view: adding dCache

http://www.pnfs.com/


pNFS Clients
Metadata
NFSv4.1 Server(s)
dCache
Management
...direct, parallel data paths...
Storage
Block (FC) • Object (OSD) • File (NFS)

Disclaimer:

pNFS here has nothing to do with the PNFS namespace provider in dCache!

… a look from the client side

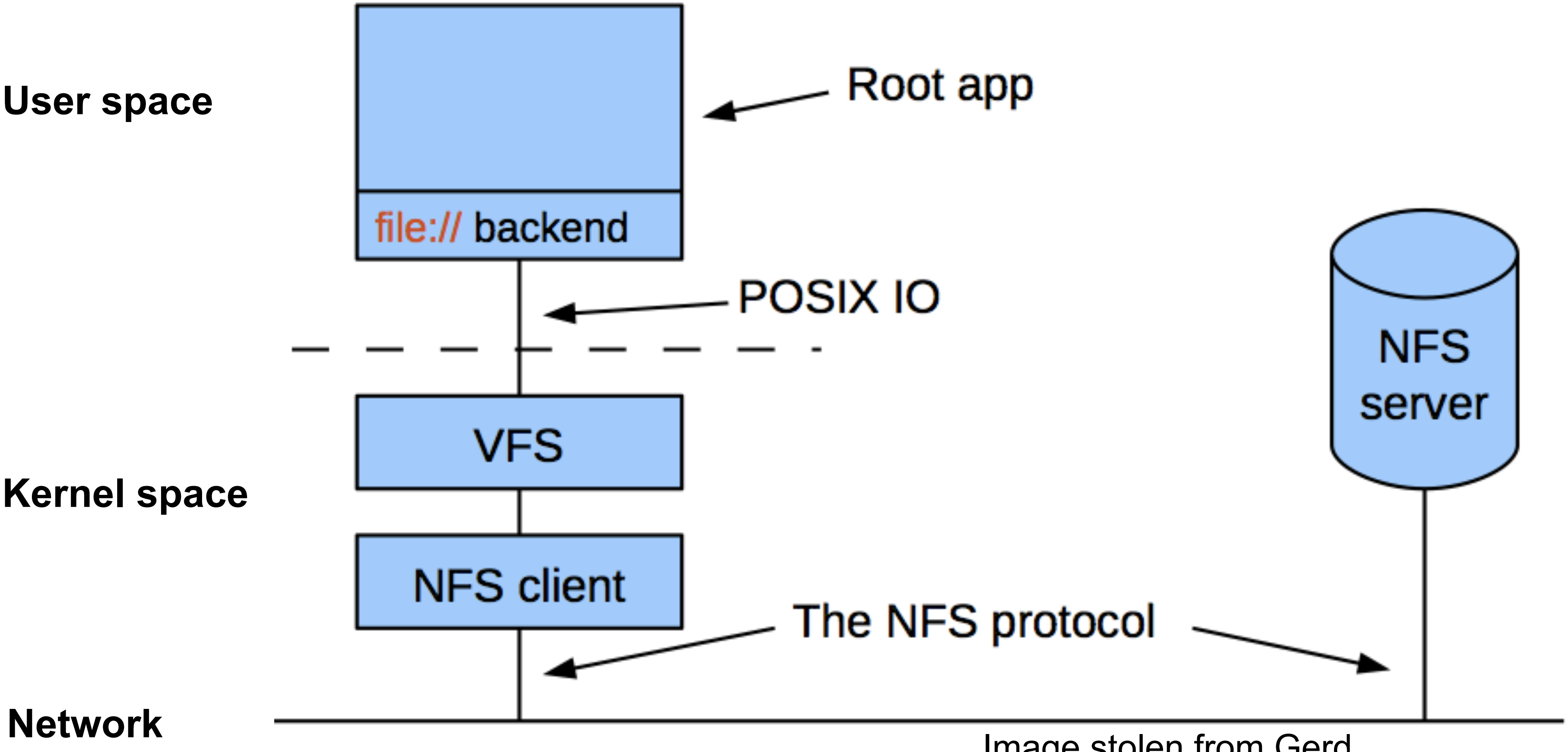


Image stolen from Gerd Behrmann

11 reasons why one should care about NFS 4.1

1) High latency link performance

   - Batching of several components, reducing number of network ops, bidirectional RPC

2) Proper authentication and authorization

   - Kerberos, X509 under investigation, ACL

3) Introduction of sessions with NFS 4.1

   - Decoupling transport from client

4) Parallel NFS (remember the plots to pages before)

5) Standardization: RFC 5661, IETF Proposed Standard

6) Industry backed: NetApp, Microsoft, Panasas, EMC, IBM, …

7) Client availability:

   - Linux (more details later), Solaris available, Windows (U.Michigan)

11 reasons why one should care about NFS 4.1 (contd)

8) Server available:

- NetApp, IBM, Oracle, EMC, IBM,…
- dCache, DPM in WLCG context

9) Clients provided by industry:

- Real POSIX IO, caching provided by OS & tuned by experts, no apps modifications

10) Funding secured

- EMI funds NFS 4.1/pNFS in DPM and dCache, HGF (D) additional funds for dCache

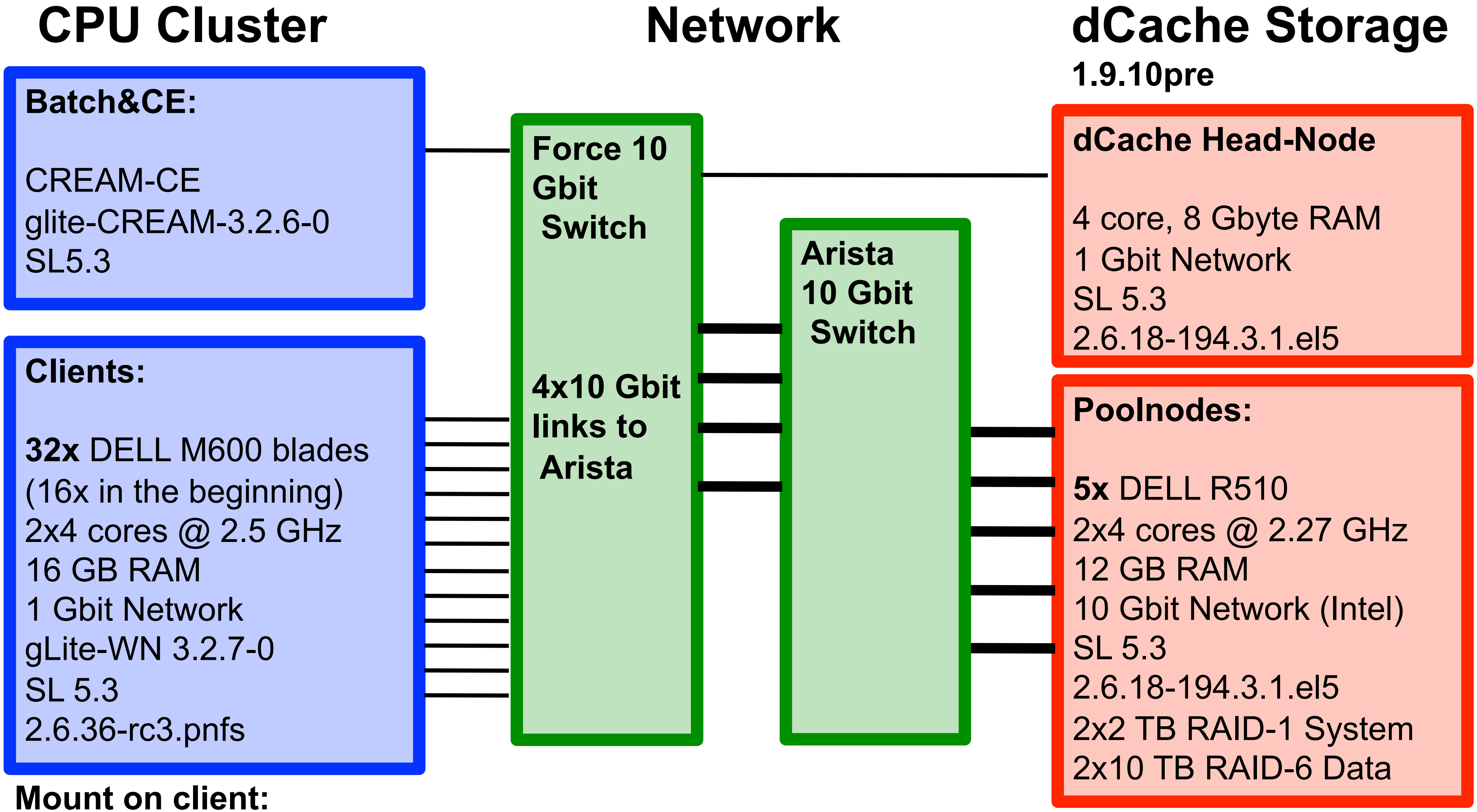
11) Simple migration path

- Server: No data migration needed, NFSv4.1 (pNFS) is additional protocol
- Clients: user file:// -> Unifies access for dCache, DPM, GPFS+Storm

OK, and how does the reality look like for HEP applications?

("11 reasons" stolen from Gerd Behrmann)

Evaluation: The testbed in the DESY GridLab



CPU Cluster

Batch&CE:

CREAM-CE
glite-CREAM-3.2.6-0
SL5.3

Clients:

32x DELL M600 blades
(16x in the beginning)
2x4 cores @ 2.5 GHz
16 GB RAM
1 Gbit Network
gLite-WN 3.2.7-0
SL 5.3
2.6.36-rc3.pnfs

Network

Force 10 Gbit Switch

4x10 Gbit links to Arista

Arista 10 Gbit Switch

dCache Storage

1.9.10pre

dCache Head-Node

4 core, 8 Gbyte RAM
1 Gbit Network
SL 5.3
2.6.18-194.3.1.el5

Poolnodes:

5x DELL R510
2x4 cores @ 2.27 GHz
12 GB RAM
10 Gbit Network (Intel)
SL 5.3
2.6.18-194.3.1.el5
2x2 TB RAID-1 System
2x10 TB RAID-6 Data

Mount on client:

```
dcache-head:/pnfs on /pnfs type nfs4 (rw,minorversion=1,rsize=32768,wsize=32768)
```

What to expect from testbed?

- Maximum BW from server ➔ Clients: 40 Gbit (link between two switches)
- Maximum BW from one pool ➔ Clients (alone): Theoretical 10 Gbit
  - Measured to 5.6 Gbit/s using iperf
- Maximum BW from Disk RAID ➔ local /dev/null
  - Measured between 520 MByte/s (few streams) and ~300 MBytes/s (random read)
- So, maximum bandwidth from Server-Disks ➔ Network ➔ Client /dev/null
  - Something between 1.5 GByte and 2.5 GByte/s
  - 32x1-Gbit clients can saturate this
- CPU ~ ½ Tier-2 whereas Storage ~¼ Tier-2
  - Clients able to really stress the storage system
  - **Storage undersized (on purpose!)**

First simple test

> Simple I/O

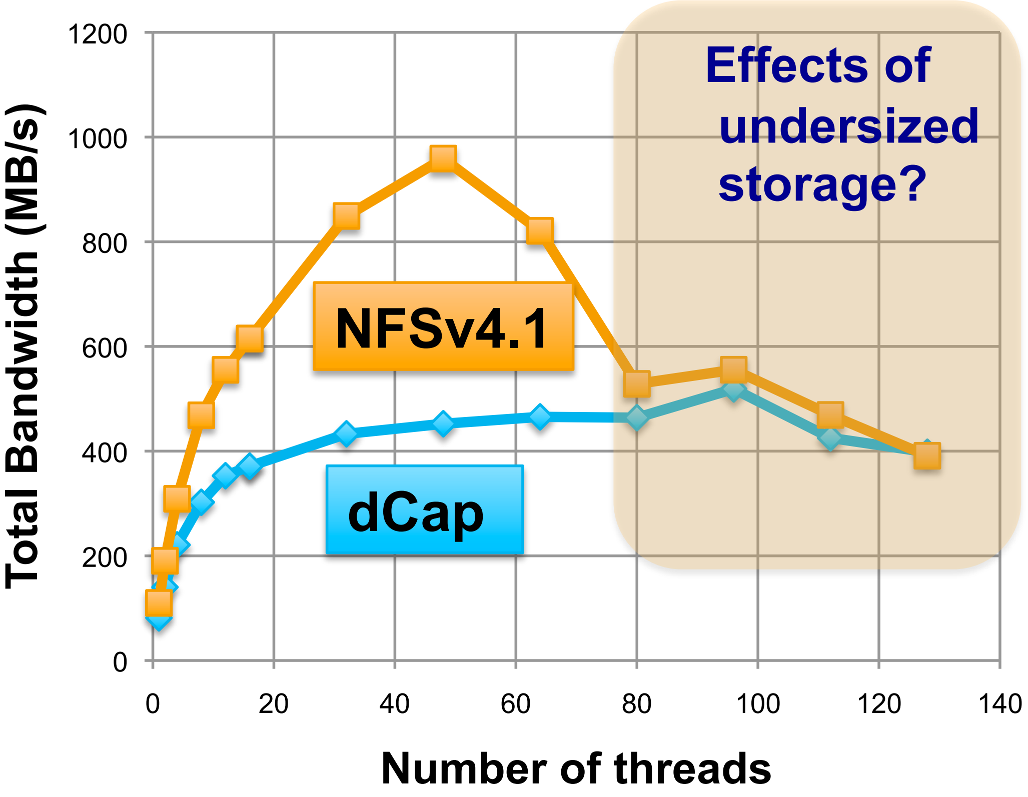
- Reading file to /dev/null
- No caching (read once, not jumping around in file)
- A maximum of 128 clients (16 nodes)

> NFS behaves better than dCap up to a certain limit

> We have no definite answer for this effect, suppose congestion on the server

- Probably due to undersized storage

→**Needs further investigation**



Stability tests

- Untaring the Linux Kernel into NFS 4.1 ✓
  - up to 16 parallel jobs (only 16 clients)
  - Works, slowly, but no problems observed with recent kernels
- CFEL Production Transfers from SLAC to DESY ✓
  - 13 TBytes over 10 days
  - 100 GBytes average file size
  - No crash
- High-Latency test: “recursive ls –l” 60k files over DSL from home ✓
  - Slow, but works
- 128 clients simultaneously writing into same file (by mistake) ✗
  - Client nodes got stuck
  - Server OK
- Clients got stuck once during ROOT tests, needed reboot ✗

ATLAS HammerCloud test: The setup

> The Data:

- Official ATLAS MC samples (7 TeV, prefereably no minbias, few jets)
- AODs, reconstructed with athena 15.6.8
- 33 TB data in total

> The Analysis

- standard AOD analysis reading Trigger and many Muon variable
- Athena 15.6.6, ROOT 5.22/00h (no ttreecache reading used)

> Initial difficulties:

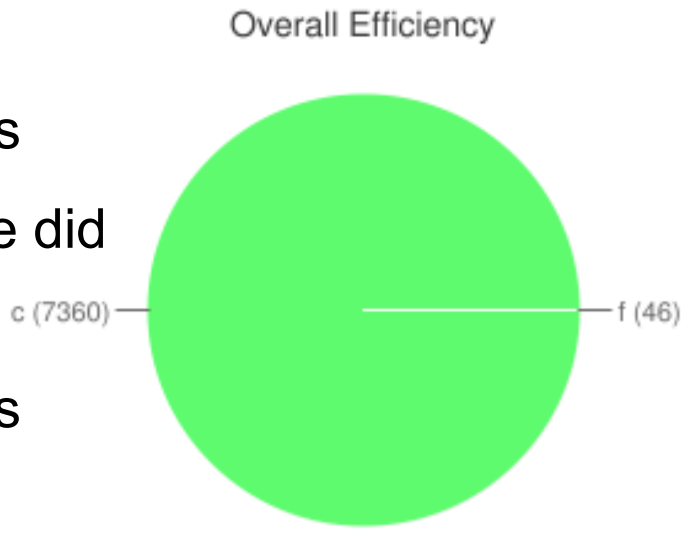
- CREAM-CE not visible, neither in Information System, nor “in the Cloud”
- dCache not a fully Grid-SE, had to provide file lists as input

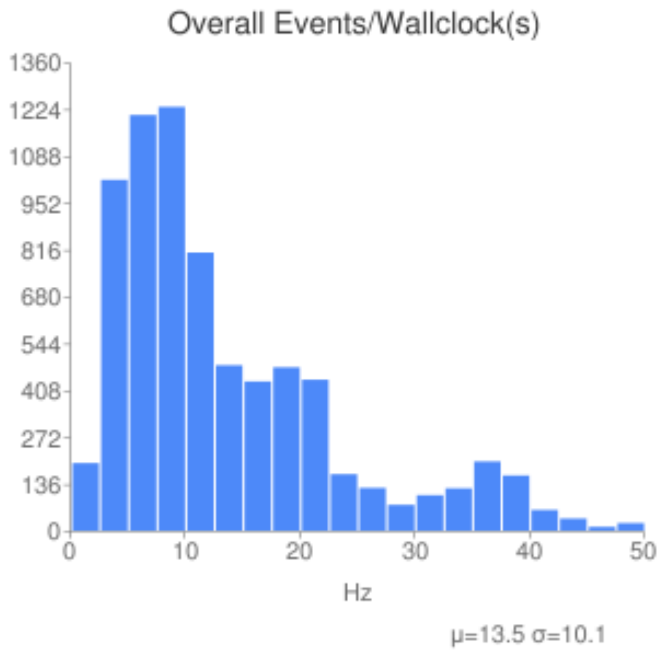
> More on HammerCloud

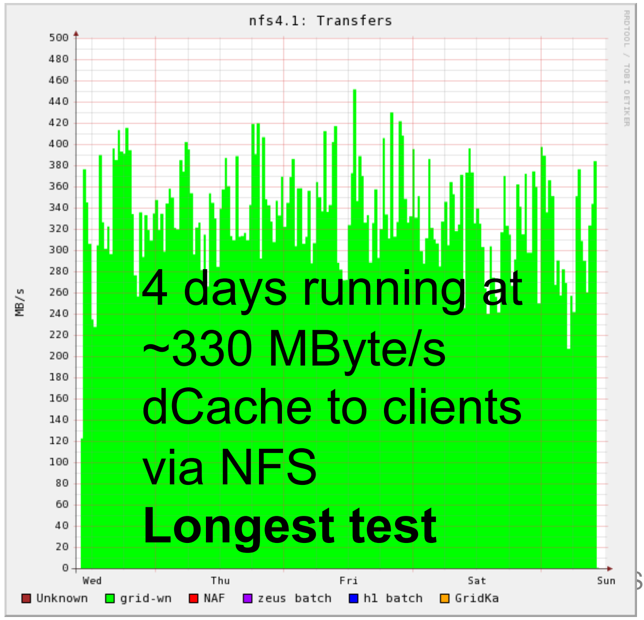
- This is the standard ATLAS application to test the performance of sites
- Parallel session 36, Dan van der Ster
- Poster PO-MON-036, Federica Legger

ATLAS HammerCloud test: The results

- 8248 jobs in total
- Cancelled after 4 days
- Longest single test we did
  - No trouble during test
- Reasonable outcomes (events/s,…)
- No comparison made to dCap (yet)



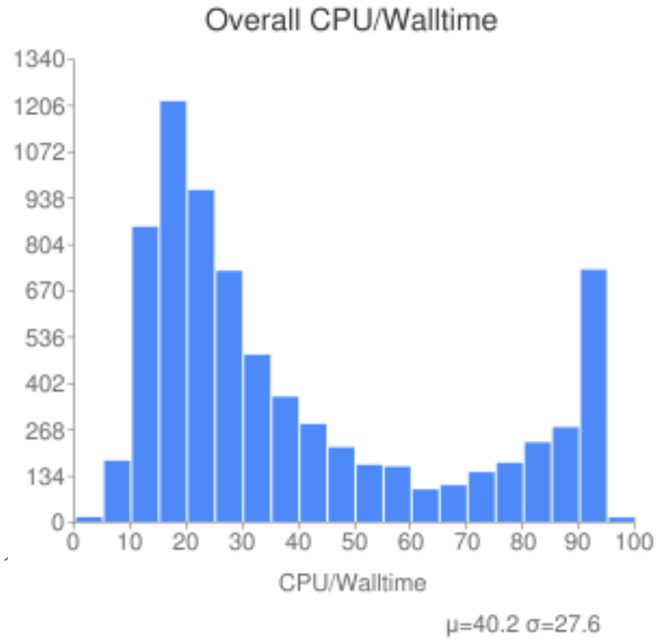




4 days running at ~330 MByte/s dCache to clients via NFS

Longest test

Sv4.



CMS Analysis: Setup

- Job submission done via the Grid and *grid-control*
  - Ability to freely define CE (which was “hidden” in our case)
  - Make use of “private” SE: Custom manipulation of the CMS Trivial File Catalogue
  - https://ekptrac.physik.uni-karlsruhe.de/trac/grid-control
- Muon analysis. Dataset: 1.7 TB in 308 RECO files
- Executable: filestest is stripping into PAT Ntuple out of the CMSSW framework
  - Using 5.22 ROOT version shipped with CMSSW
- One typical use-case on the DESY National Analysis Facility
- Not much CPU, nearly only I/O
- Evaluation of performance metrics in CMSSW framework job report (Andrzej Wronka (summerstudent at DESY))

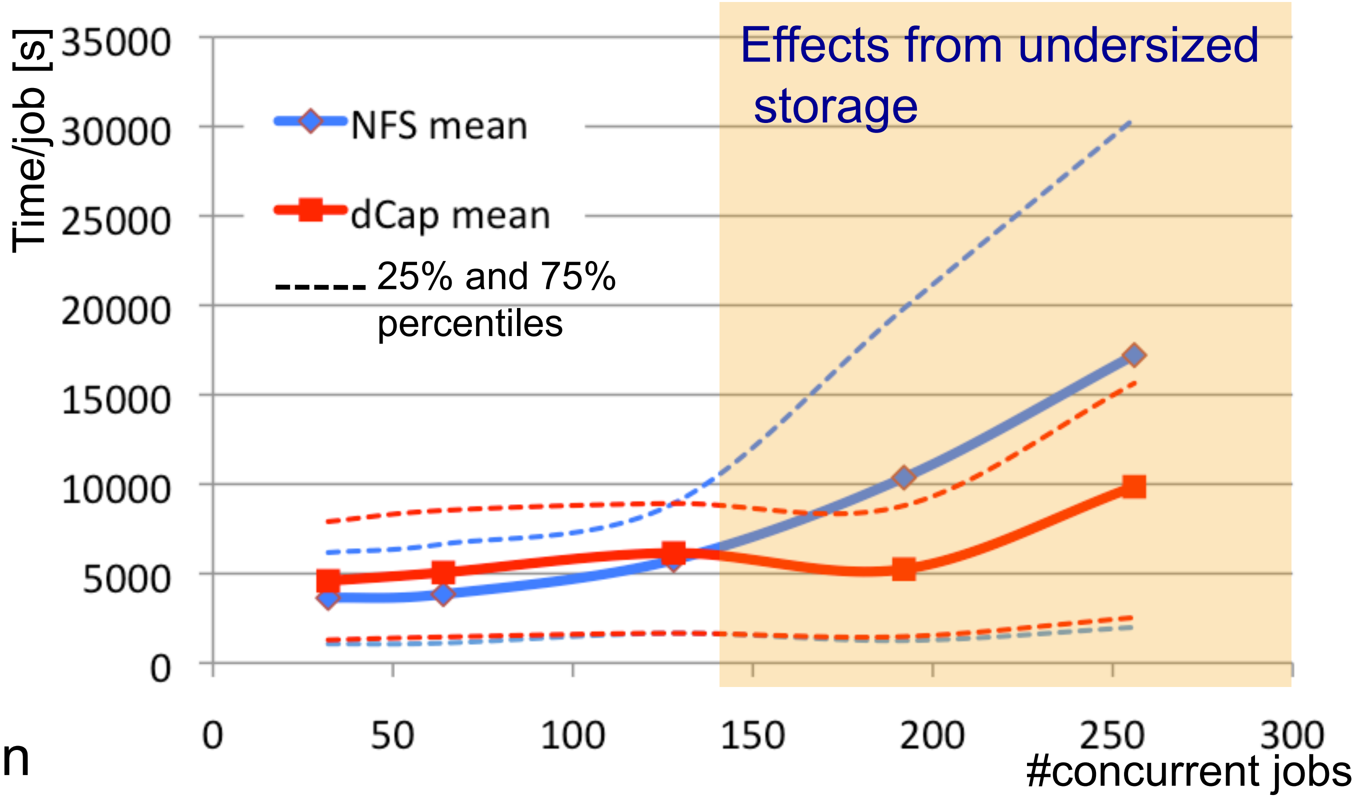
CMS Analysis: Results

Below ~128 jobs:

- > NFS 20% faster than dcap

Above ~128 jobs:

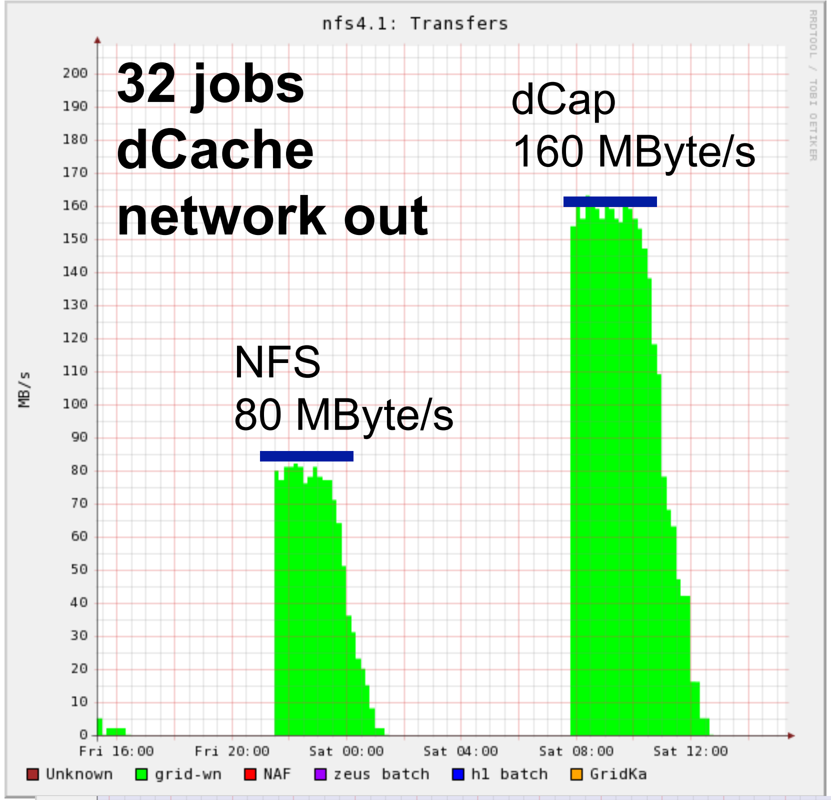
- > NFS performance degrades, dcap only slightly degrades
- > Not yet fully understood, suspect numbers of threads in dCache NFS server
- > Checked that client congestion not fault



Effects of File system cache:

- > dCap reads 2.5 times more data than NFSv4.1 (dCache billing logs and network monitoring plots): Next slide:

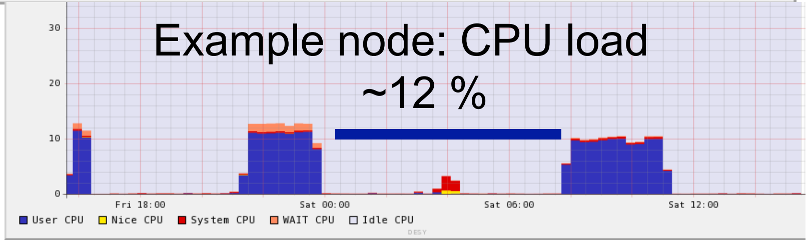
CMS Tests: A look at dCache and one node



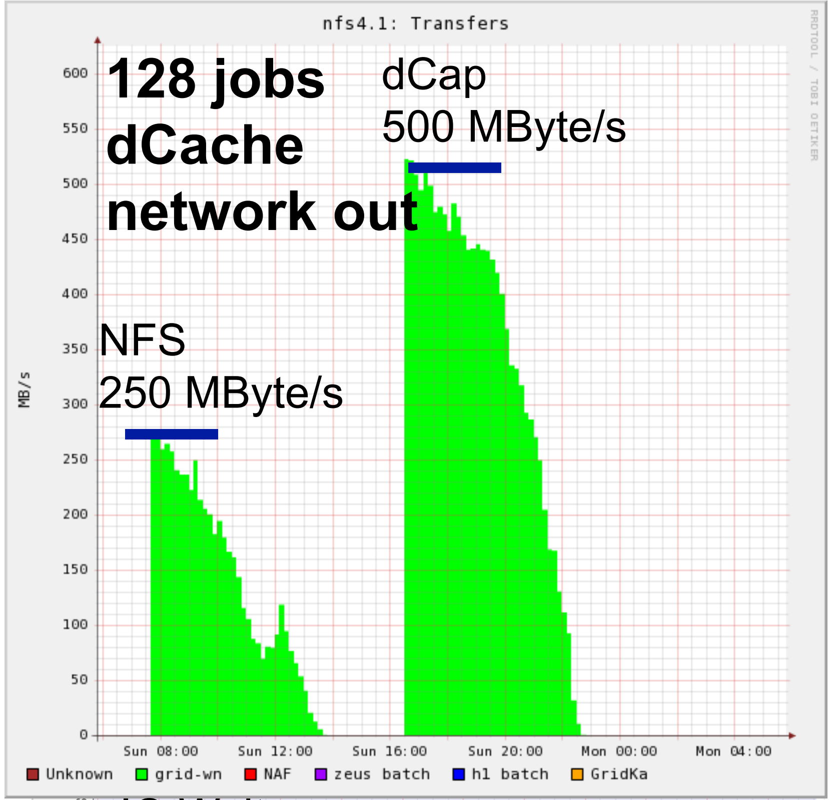
32 jobs dCache network out

NFS 80 MByte/s

dCap 160 MByte/s



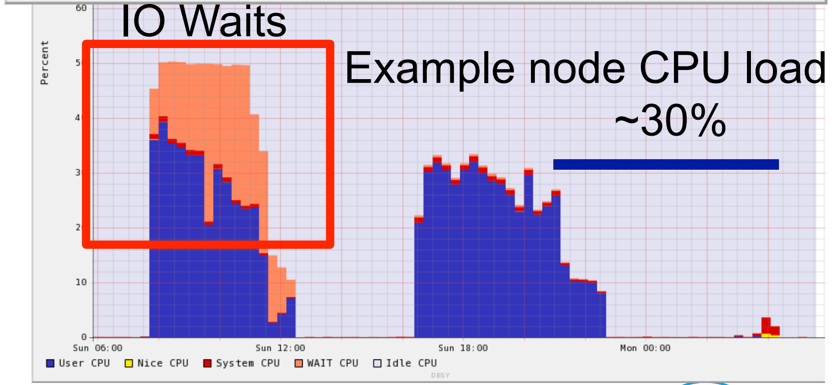
Example node: CPU load ~12 %



128 jobs dCache network out

NFS 250 MByte/s

dCap 500 MByte/s



IO Waits

Example node CPU load ~30%

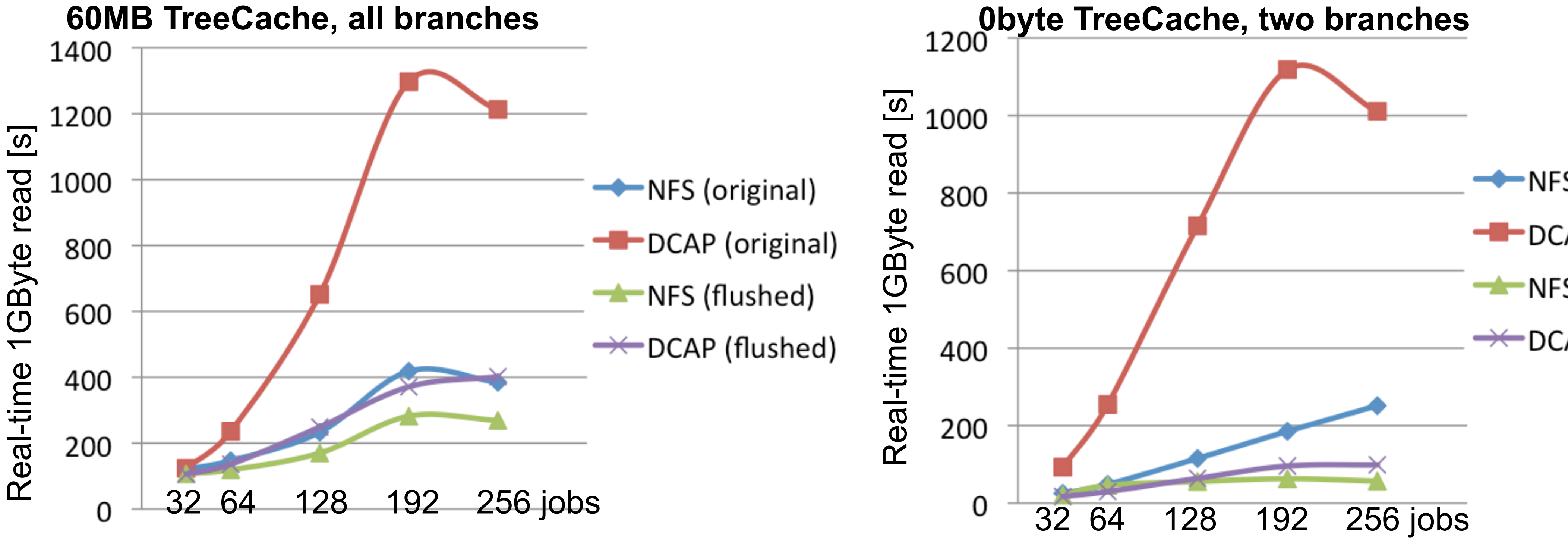
- IO waits gets more important for NFS at higher numbers of concurrent jobs
- Less network traffic for NFS

Half-Synthetic ROOT tests: Setup

- New ROOT version 5.27.06, compiled with dCap support
- Files provided by René Brun: `atlasFlushed.root` (re-organized files with optimized buffers) and `AOD.067184.big.pool_4.root` (some other original file) (flushed: 1GByte, original 1.3 GByte)
- Test script provided by René: simple script reading events: `taodr.C`
- Different test runs:
  - Reading via NFS or dCap
  - Reading with 60MByte TreeCache, or with 0Byte TreeCache
  - Reading all branches or only 2 branches
  - 32, 64, 128, 192 or 256 jobs running in parallel

- Last minute-result! Have not spoken with ROOT people!

Half-Synthetic ROOT tests: Results



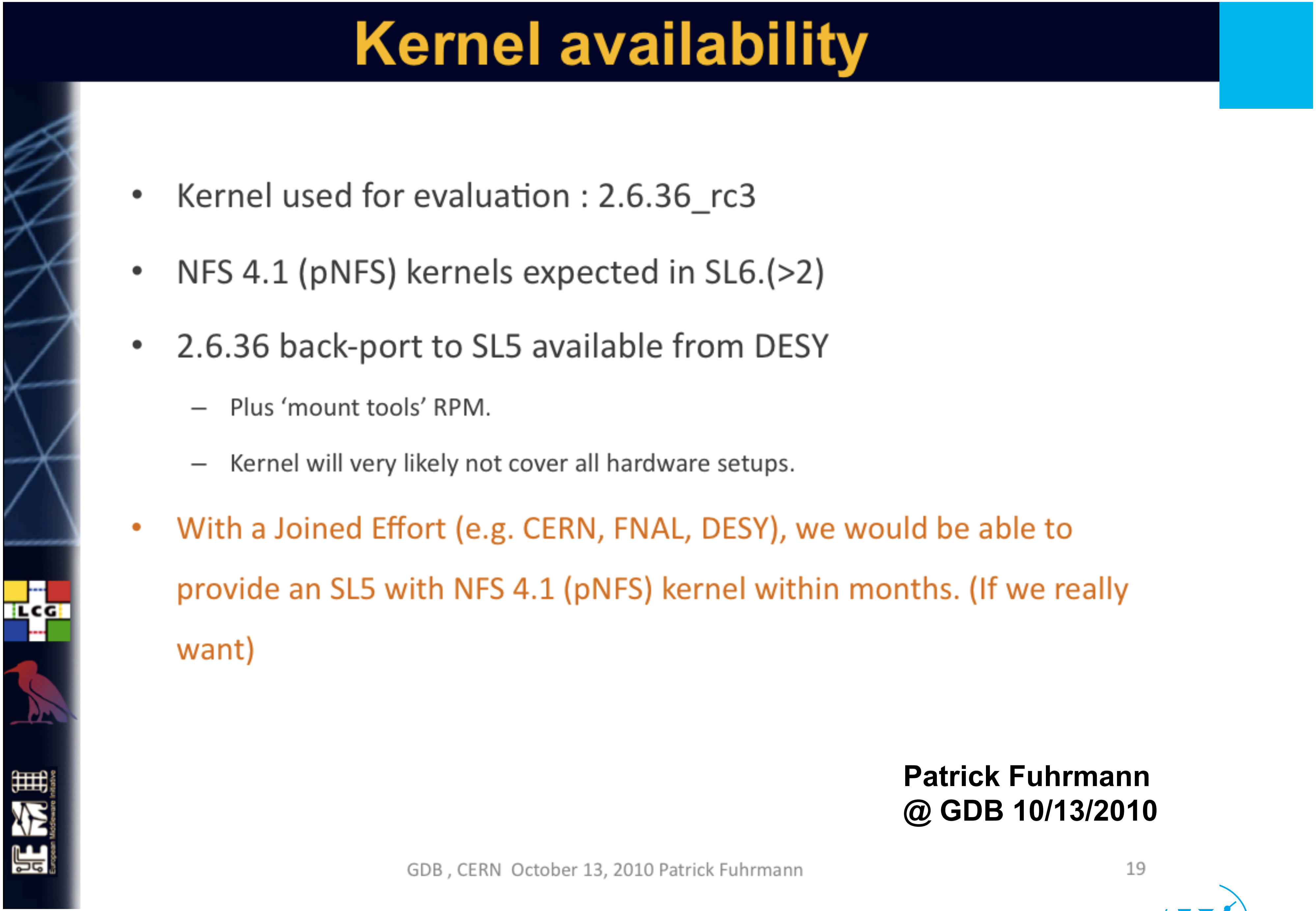
> NFS better for original and flushed files than dCap

  - Flushed: not much difference, original: Large difference

> TreeCache helps, NFS adds additional speed

> Peak at 192 clients not understood

> Remember: Just going through events and doing nothing … not really representative for analysis



Kernel availability

- Kernel used for evaluation : 2.6.36_rc3
- NFS 4.1 (pNFS) kernels expected in SL6.(>2)
- 2.6.36 back-port to SL5 available from DESY
  - Plus ‘mount tools’ RPM.
  - Kernel will very likely not cover all hardware setups.
- With a Joined Effort (e.g. CERN, FNAL, DESY), we would be able to provide an SL5 with NFS 4.1 (pNFS) kernel within months. (If we really want)

Patrick Fuhrmann
@ GDB 10/13/2010

Summary

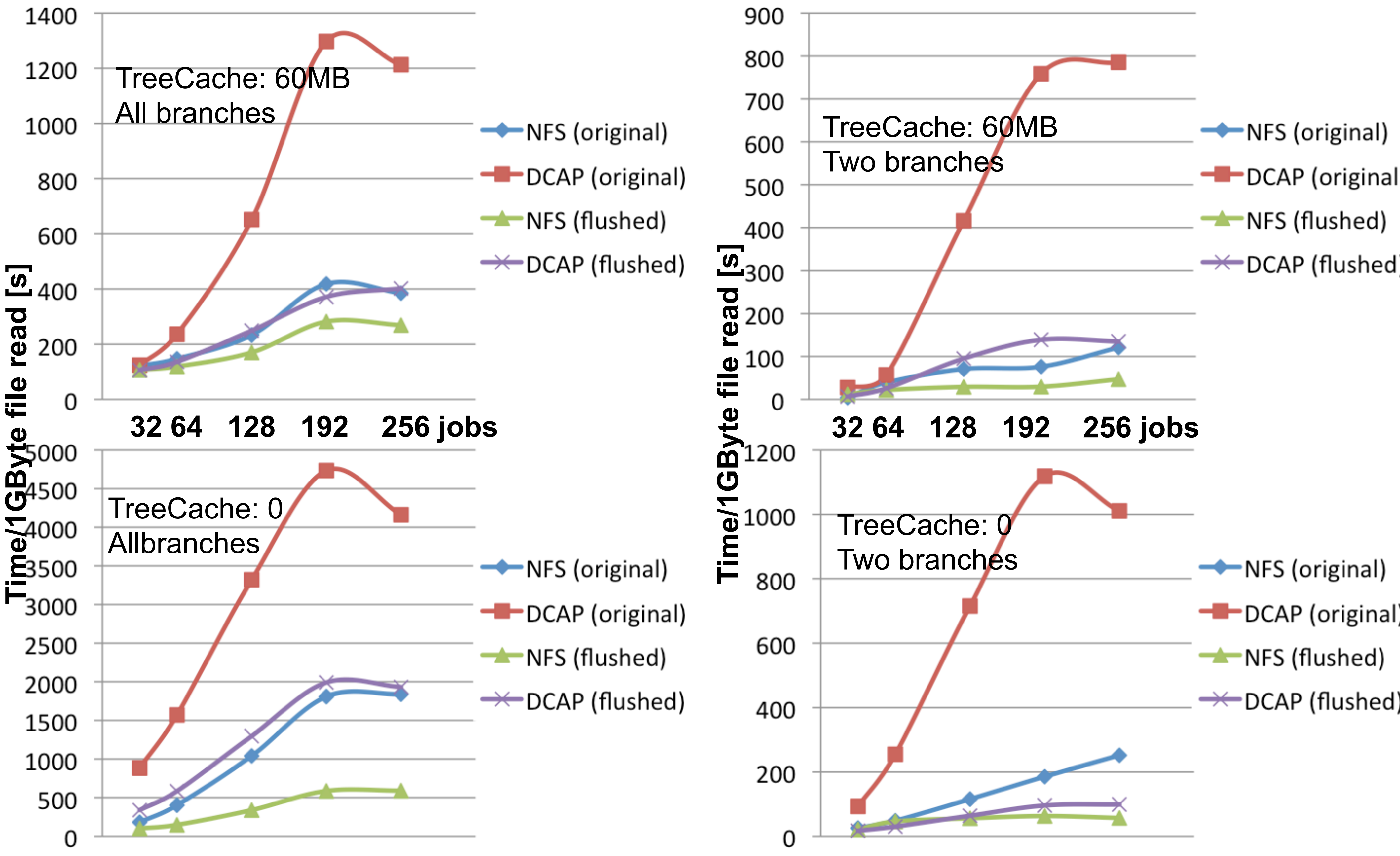
- Set up different use cases
  - Synthetic, ATLAS HammerCloud, CMS analysis, ROOT files
  - No change to experiments applications needed
  - Managed to be run and steered by non-experts (like me)
- Set up a test bed comparable to a small Grid site
  - **Underpowered w.r.t dCache storage**: Able to see bottlenecks
- Presented results
  - **Synthetic**: Provide general performance and stability measurements of NFS 4.1/pNFS
  - **ATLAS HammerCloud:** Stable and well-performing running over four days
  - **CMS analysis:** See effects of FS cache, excellent behavior of NFS up to some point
  - **ROOT files**: See effects of FS cache, better performance than dcap, even with most recent ROOT version and with TreeCache enabled

- **NFS 4.1/pNFS has advantages over traditional proprietary protocols**
- **We now know: Performance is one of them!**

Future

- More tests needs to be done, some issues have to be understood and fixed
- Remember: NFS4.1 (pNFS) is not dCache only. NetApp have promised to give us a test storage a.s.a.p. (unfortunately not in CHEP timeline…)
  - DPM: Talk by Ricardo Rocha in Parallel Session 15
- No mentioning of security, authentication, authorization here. This needs to come next (and will!)
- Maybe it is time to think about a backport of NFS 4.1 (pNFS) into SL5 kernel? Could this be a combined effort? Would be a temporary effort!

Backup 1: Complete set of ROOT result plots


Time/1GByte file read [s]
TreeCache: 60MB
All branches
32 64 128 192 256 jobs
◆ NFS (original)
■ DCAP (original)
▲ NFS (flushed)
× DCAP (flushed)
TreeCache: 60MB
Two branches
32 64 128 192 256 jobs
◆ NFS (original)
■ DCAP (original
▲ NFS (flushed)
× DCAP (flushed
TreeCache: 0
Allbranches
◆ NFS (original)
■ DCAP (original)
▲ NFS (flushed)
× DCAP (flushed)
Time/1GByte file read [s]
TreeCache: 0
Two branches
◆ NFS (original)
■ DCAP (original)
▲ NFS (flushed)
× DCAP (flushed)